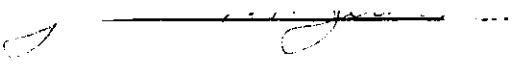
In presenting the dissertation as a partial fulfillment of the requirements for an advanced degree from the Georgia Institute of Technology, I agree that the Library of the Institute shall make it available for inspection and circulation in accordance with its regulations governing materials of this type. I agree that permission to copy from, or to publish from, this dissertation may be granted by the professor under whose direction it was written, or, in his absence, by the Dean of the Graduate Division when such copying or publication is solely for scholarly purposes and does not involve potential financial gain. It is understood that any copying from, or publication of, this dissertation which involves potential financial gain will not be allowed without written permission.



7/25/68

SOME ALGORITHMS FOR NON-LINEAR REGRESSION PROBLEMS

A THESIS

Presented to

The Faculty of the Graduate Division

by

James M. Jerkins

In Partial Fulfillment

of the Requirements for the Degree

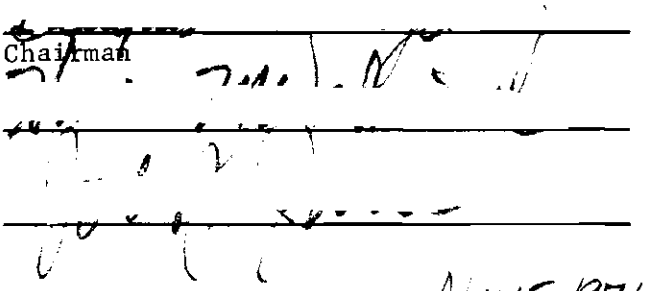
Master of Science in Industrial and Systems Engineering

Georgia Institute of Technology

December, 1971

SOME ALGORITHMS FOR NON-LINEAR REGRESSION PROBLEMS

Approved:



Chairman

Date approved by Chairman: NOV 15, 1971

TABLE OF CONTENTS

LIST OF TABLES

SUMMARY

The existing methods of solving non-linear regression problems are presented and discussed. These include the linearization, steepest descent, and Marquart's compromise methods. A new method for finding the minimum of a sum of squares function used in non-linear regression problems is described and compared to the compromise method of D. W. Marquart.

The new method is called the Modified Sequential Simplexalgorithm and illustrates the advantage of not requiring the calculation of derivatives while being extremely easy to program. The major modification included in this algorithm is the attempted optimization of the step length at each iteration.

Although the Modified Sequential Simplex Algorithm is simpler and performs fewer calculations per iteration, it converged only slightly slower than Marquart's compromise method as illustrated by a comparison on a UNIVAC 1108 computer. The results prove the feasibility of the method and illustrate some possible modification.

CHAPTER I

INTRODUCTION

Statement of the Problem

A response Y is supposed dependent on the levels of K quantitative factors or variables $\underline{X}^T = \lfloor X_1, X_2, \cdots, X_K \rfloor$ which are capable of accurate measure and possibly control. For the uth combination of factor levels ($u = 1, 2, \ldots, m$), then

$$Y_u = \emptyset\ (X_{1u}, X_{2u}, \ldots, X_{Ku}) + \epsilon_u \qquad (1)$$

where ϵ_u is the error associated with this uth combination and measurement. The problem is to fit a non-linear model to represent $\emptyset$ over all possible combinations of the factor levels that are of interest. To find this representation, one uses the m combinations or observations ($u = 1, 2, \ldots, m$) consisting of K+1 terms each, say

$$Y_u, X_{1u}, X_{2u}, \ldots, X_{Ku}.$$

Given a mathematical model of the form $Y = f(\underline{X}, \underline{\theta})$ where $\underline{\theta}^T = \lfloor \theta_1, \theta_2, \ldots, \theta_p \rfloor$ are the parameters in the model, we wish to use the observed data to find a vector $\underline{\theta}$ which will bring $f(\underline{X}, \underline{\theta})$ as close to $\emptyset(\underline{X})$ as possible. That is, we wish to fit the model

$$Y_u = f(\underline{X}_u, \underline{\theta}) + \epsilon_u,\ u = 1, 2, \ldots, m. \qquad (2)$$

Least Squares Criteria

As stated in the previous section, the problem is to fit the model

$$Y = f(\underline{X}, \underline{\theta}) + \epsilon \tag{3}$$

Almost the only criterion used today is least squares. That is, the smaller the sum of the squares of the difference $(Y_u - f(\underline{X}_u, \underline{\theta}))$ for $u = 1, 2, \ldots, m$ the truer or closer the model fits the data. This is based on the assumption that the data represents the true function $\emptyset$ within acceptable measurement error.

By minimizing the error sum of squares function

$$F(\theta) = \sum_{u=1}^{m} [Y_u - f(\underline{X}_u, \underline{\theta})]^2 \tag{4}$$

over all possible θ, one obtains the best model fit. If we assume the normality and independence of errors, then $\underline{\epsilon} = N(\underline{0}, \underline{I}\,\sigma^2)$ where

$$\underline{\epsilon}^T = (\epsilon_1, \epsilon_2, \ldots, \epsilon_u) \tag{5}$$

$\underline{0}$ is a vector of zeros and $\underline{I}$ is a unit matrix, both of appropriate sizes. It then follows that the least squares estimate of θ (say $\underline{\theta}^*$) is also the maximum likelihood estimate of θ. This is because the likelihood function is

$$h(\underline{\theta}, \sigma^2) = (2 \quad \sigma^2)^{-m/2} \; e^{-F(\theta)/2\sigma^2} \tag{6}$$

so that if σ^2 is known, maximizing $h(\underline{\theta}, \sigma^2)$ is equivalent to minimizing $F(\theta)$ with respect to $\underline{\theta}$.

Scope and Purpose of Thesis

The problem of finding the parameters of a physical theory or model from experimental results is frequently reduced to finding the minimum of an error sum of squares of a non-linear function. If derivatives are available, then there are highly efficient and simple ways described in Chapter II of this thesis which may be used. The Marquart compromise algorithm is probably the most widely used procedure. However, in many cases it is impossible (or nearly so) to calculate gradients, and for those problems algorithms based on the sequential simplex procedure are of great use as they require no derivatives. The purpose of this thesis is to provide a listing and description of the major methods of non-linear regression analysis including a sequential simplex algorithm developed for this thesis called the Modified Sequential Simplex Algorithm, which attempts to optimize the step length at each iteration. Also presented is a comparison over a series of problems of the Modified Sequential Simplex Algorithm and Marquart's compromise algorithm.

CHAPTER II

EXISTING ALGORITHMS

The literature concerned with non-linear regression contains several algorithms for the solution of the problems encountered. This chapter is a survey of those existing methods.

Linearization

The linearization algorithm involves the repeated linear approximization of certain non-linear functions for its solution. As shown in the discussion of least squares in Chapter I, one wishes to find a vector $\underline{\theta}^T = [\theta_1, \theta_2, \ldots \theta_p]$, that minimizes the error sum of squares function:

$$F(\theta) = \sum_{u=1}^{m} (Y_u - f(\underline{X}_u, \theta))^2 \tag{1}$$

Using the notation

$$\sum_{u=1}^{m} [g_u(\underline{\theta})]^2 = \sum_{u=1}^{m} (Y_u - f(\underline{X}_u, \underline{\theta}))^2 \tag{2}$$

and also

$$g_u^i(\underline{\theta}) = \frac{\delta}{\delta\theta_i} g_u(\underline{\theta}) \tag{3}$$

$$g_u^{ij}(\underline{\theta}) = \frac{\delta^2}{\delta\theta_i \delta\theta_j} g_u(\underline{\theta}) \tag{4}$$

for the first and second derivatives of the m different functions involved in the sum of squares function, the method becomes more easily understandable.

Each iteration of the algorithm requires an approximation $\underline{\theta}'$ to the true minimizing vector. This vector approximation is not likely to give the true minimum value for the sum of squares function, but the true minimum will be given by some vector $\underline{\theta}' + \underline{\varepsilon}$. By differentiating (1) one obtains

$$\sum_{u=1}^{m} g_u^i(\underline{\theta}' + \underline{\varepsilon}) \cdot g_u(\underline{\theta}' + \underline{\varepsilon}) = 0_j \quad i = 1, 2, \ldots, p \tag{5}$$

The next step in the algorithm is to approximate

$$\sum_{u=1}^{m} g_u^i(\underline{\theta}' + \underline{\varepsilon}) \cdot g_u(\underline{\theta}' + \underline{\varepsilon}) \tag{6}$$

by the first two terms of the Taylor series in $\underline{\varepsilon}$ about $\underline{\theta}'$, giving

$$\sum_{u=1}^{m}\left[g_u^i(\underline{\theta}') \cdot g_u(\underline{\theta}') + \sum_{j=1}^{p}(g_u^{ij}(\underline{\theta}')g_u(\underline{\theta}') + g_u^i(\underline{\theta}') \cdot g_u^j(\underline{\theta}'))\varepsilon_j\right]=0 \tag{7}$$

A further approximation must now be made. The development of this algorithm depends on the term

$$g_u^{ij}(\underline{\theta}') \cdot g_u(\underline{\theta}') \tag{8}$$

being small enough to ignore. Some discussion of this will be made later in this section. The iteration may now be completed by calculating the correction to $\underline{\theta}'$ by solving for $\underline{\varepsilon}$ in

$$\sum_{j=1}^{p}\left[\sum_{u=1}^{m} g_k^i(\underline{\theta}') \cdot g_u^j(\underline{\theta}')\right]\varepsilon_j = -\sum_{u=1}^{n} g_k^i(\underline{\theta}') \cdot g_u'(\underline{\theta}') \tag{9}$$

$$i = 1, 2, \ldots, p$$

A check is then made to see if the new vector $\underline{\theta}' + \underline{\varepsilon}$ satisfies whatever termination criterion is in use and the algorithm terminates or begins another iteration.

A deviation from this basic method has been suggested by Powell (Powell, 1965) in which no derivatives are required and which he claims had convergence comparable to the algorithm just described. In the above method, convergence hinges on the size of the second derivative term (8). This term is of order $\underline{\varepsilon}$ if $g_u(\underline{\theta}')$ is zero at the minimum sum of squares, and disappears if $g_u(\underline{\theta})$ is linear in all the variables. For any other case the convergence of the procedure is linear. Powell's method requires, in addition to an initial estimate of the minimum vector, p linearly independent directions in the space of the variables together with estimates of the derivatives of the g_u along with the directions. The algorithm then proceeds to find an $\underline{\varepsilon}$ using the estimated derivatives and a λ_m to minimize $F(\underline{\theta}' + \lambda\underline{\varepsilon})$. The approximation of the minimum vector $\underline{\theta}$ is then replaced by $\underline{\theta}' + \lambda_m \varepsilon$ and the next iteration may be begun.

H. O. Hartley has presented a method (Hartley, 1961) which modifies the basic linearization method to guarantee convergence. He makes two conditions or assumptions to guarantee convergence. The first assumption is that for any non-trivial set of $u_i(1, 2, \ldots, p)$ with $\sum u_i^2 > 0$,

$$\sum_{u=1}^{m} \left(\sum_{i=1}^{p} u_i \, g_u^i(\underline{\theta}') \right)^2 > 0 \qquad (10)$$

for all $\underline{\theta}'$ in a bounded convex set S of the parameter space $\theta_1, \theta_2, \ldots \theta_p$. The second assumption is the one which Hartley uses to guarantee

convergence. He denotes

$$F = \lim_{\bar{S}} \inf F(\theta) \tag{11}$$

where $\bar{S}$ is the complement to S, then he assumes it possible to find a vector $\underline{\theta}'$ in the interior of S such that

$$F(\underline{\theta}') < F \tag{12}$$

The algorithm then proceeds in the familiar manner of computing corrections to the elements of the starting vector by replacing the function described above with a Taylor expansion and solving for a correction vector $\underline{\varepsilon}$.

The linearization techniques have two major shortcomings. First, they require the existence of both first and second derivatives, and secondly there are many non-linear regression equations which just cannot be adequately approximated by the first two Taylor series terms. For these reasons, linearization procedures often converge extremely slowly.

Steepest Descent

The steepest descent or gradient method is a widely applicable algorithm for certain classes of problems not handled easily by faster algorithms. Among these problems are those for which constraints on the variables exist, those for which there are so many variables that more complicated methods exceed the computer storage, and those for which the computed values are subject to error making derivative values of the second order questionable. Standard steepest descent

methods do not require the algorithm to compute or approximate second derivatives and do not require the location of an optimum along a line. These are various modifications to the basic algorithm often employed to obtain faster convergence. Several of these modifications will be discussed later in this section.

The rationale for the steepest descent algorithm is that if moves are always made in the direction the function is decreasing most rapidly, and the steps taken are small enough, then the value of the function will always decrease. Eventually the minimum will be reached and the algorithm must stop. At some given starting point the direction of the most rapid decrease in function value $\underline{\delta}_g$ is calculated and the algorithm takes a step of predetermined length in that direction. The value of the sum of squares function is calculated to insure it has been decreased and a new gradient direction is found. A step is made in the new direction and the iterations continue until some termination criterion has been satisfied.

This very straightforward procedure has been changed and modified many times by various authors in an attempt to obtain faster convergence. Ramsay (Ramsay, 1970) describes a slight change in the basic algorithm which he calls the gradient path method. In the limit, as the size of the steps in the algorithm get smaller, the steepest descent path is defined beginning at some point in the parameter space and passing through the minimum. Let some initial point be $\underline{p}$ and define the gradient of some arbitrary point on the path, $\underline{X}$, to be $s(\underline{X})$. The arc length along the path from $\underline{p}$ to $\underline{X}$ is

called r. The point $\underline{X}$ can now be defined as a function of arc length:

$$\underline{X} = y(r) \tag{13}$$

The tangent of the path $y'(r)$ lies along the gradient vector and can be calculated by

$$y'(r) = \frac{s(y)}{\| s(y) \|} \tag{14}$$

and the problem reduces to a set of simultaneous differential equations. The equation for the tangent (14) is accurate because the tangent of a path defined as a function of arc length has unit length.

Fletcher and Powell (Fletcher and Powell, 1963) found it convenient to group algorithms into two classes according to whether the gradient vector is defined analytically at each point or must be estimated from the differences of values of the function. They describe a method with rapid convergence due to the use of an approximated matrix of second derivatives. Powell's algorithm (Powell, 1965), although slightly different, also uses the matrix of approximated derivatives, as does the method of Fletcher and Reeves (Fletcher and Reeves, 1964).

Powell (Powell, 1962) discusses a method with second-order convergence which does not use a matrix of second derivatives. He defines second-order convergence as a property of an algorithm such that if and only when $g(\underline{X})$ in equation 15 equals $f(\underline{X})$ then the method goes from some arbitrary value to the minimum vector in one cycle. That is, with the function being $f(X)$, the number of independent variables being n, and the minimum vector being $\underline{\epsilon}$, then $g(X)$ is defined

as

$$g(\underline{X}) = f(\underline{E}) + \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (X_i - E_i)(X_j - E_j) \frac{\delta^2}{\delta X_i \circ X_j} f(\underline{E}) \qquad (15)$$

The algorithm that Powell describes hinges on a corollary of the theorem that because $g(\underline{X})$ is quadratic in the independent variables any line which passes through $\underline{E}$ intersects the members of a family of contours $f(\underline{X}) = c$ at equal angles. The corollary states that if the normal to the contour $f(\underline{X}) = f(\underline{t})$ at some point $\underline{t}$ is parallel to the normal to the contour $f(\underline{X}) = f(\underline{t}')$ at some point $\underline{t}'$, then the line which joins $\underline{t}$ and $\underline{t}'$ passes through $\underline{E}$.

Powell's algorithm proceeds from an initial estimate of the minimum vector to a point $\underline{E}$ along a line which is normal to the contour $f(\underline{X}) = f(\underline{\tau})$ and passes through $\underline{\tau}$. The point $\underline{E}$ is found where the derivative of $f(\underline{X})$ with respect to the distance along the line is zero. Then the method finds the stationary value of $f(\underline{X})$ in the (n-1) dimentional hyperplane which contains $\underline{E}$ and some point $\underline{\delta}$ in the hyperplane is perpendicular to the line joining $\underline{E}$ to $\underline{\tau}$. Then, since the normal to $f(\underline{X}) = f(\underline{\delta})$ at $\underline{\delta}$ is parallel to the normal at τ, we now have a line on which the new estimated minimum exists where derivative of $f(\underline{X})$ with respect to the distance along the line is zero.

At points away from the minimum this method is close to the basic method of steepest descent. However, this method will converge much faster near the minimum.

Marquart's Compromise Procedure

The two previously discussed approaches to non-linear estimation have served as the basis of most algorithms for non-linear regression. In this section, Marquart's compromise algorithm will be discussed, following the presentation by Marquart (Marquart, 1963). The method of linearization and the method of steepest descent often encounter difficulty and fail to converge. The steepest descent method usually failsbecause of its extremely slow convergence after the first few iterations and the linearization method usually fails because of divergence of successive iterations.

In minimizing the sum of squares function

$$F(\underline{\theta}) = \sum_{u=1}^{m} (Y_u - f_u(\theta))^2 \tag{16}$$

the iterative procedure begins with an estimate of the minimum vector $\underline{\theta}^*$ and passes through a series of other estimates successively. Calling these estimates $\underline{\theta}_1, \theta_2, \ldots, \underline{\theta}_j, \ldots$ and so on, we will call the elements of the jth vector $\underline{\theta}_j' = [\theta_{1j}, \theta_{2j}, \ldots, \theta_{pj}]$.

The steepest descent algorithm discussed in the previous section moves from $\underline{\theta}$ j in a direction of negative gradient at $\underline{\theta}$ j. Call that vector direction $\underline{\delta}$ g. Its components are

$$\underline{\delta}\, g = \left(-\frac{\delta F(\underline{\theta})}{\delta \theta_1}, -\frac{\delta F(\underline{\theta})}{\delta \theta_2}, \ldots, -\frac{\delta F(\underline{\theta})}{\delta \theta_p} \right) \tag{17}$$

where all the derivatives are evaluated at $\underline{\theta}$ j.

To make notation compatible with the Marquart procedure,

equation (9) must be rewritten as

$$\underline{\delta}_{\ell} = (\underline{P}_j' \underline{P}_j)^{-1} \underline{P}_j' (Y - f_j) \tag{18}$$

where

$$\underline{P}_j = \sum_{j=1}^{m} f_j^i (\underline{\theta}') \; ; \; i = 1, 2, \ldots, n \tag{19}$$

and

$$\underline{\delta}_{\ell} = \underline{E} \tag{20}$$

The linearization method moves from $\underline{\theta}$ j in a direction $\underline{\delta}_{\ell}$ determined by equation (18). Marquart's compromise procedure is a method that in some sense will interpolate between the $\underline{\delta}_g$ of the steepest descent method and the $\underline{\delta}_{\ell}$ of the linearization method. In both the steepest descent method and linearization the choice of the correction vector was made before a decision on the step size. In the algorithm forming the basis of Marquart's compromise procedure the direction and step size are determined simultaneously.

The theoretical basis for the algorithm is contained in three theorems, the first two of which are due to D. D. Morrison. The proofs of the theorems are contained in the original paper by Marquart (Marquart, 1963).

Theorem 1

Let $\lambda \geq 0$ be arbitrary and let $\underline{\delta}_o$ satisfy the equation

$$(\underline{P}_j' \underline{P}_j + \lambda \underline{I}) \underline{\delta}_o = \underline{P}_j' (Y - f_j) \tag{21}$$

then $\underline{\delta}_o$ minimizes with respect to $\underline{\theta}$

$$SS_j(\underline{\theta}) = \sum_{u=1}^{n} (Y_u - f_u(\underline{\theta}_j) - \sum_{i=1}^{P} (\theta_i - \theta_{ij}) P_{iu}^{j})^2 \qquad (22)$$

under the spherical constraint

$$|| \underline{\delta} ||^2 = || \underline{\delta}_o ||^2 \qquad (23)$$

where $|| \underline{\delta}_o ||$ is a specified radius. It should be noted that $SS_j(\underline{\theta})$ defines a sum of squares function involving the linear expansion approximating the non-linear model at $\underline{\theta}_j$ rather than the true non-linear model. Also note that $|| \delta_o ||$ denotes the length of $\underline{\delta}_o$, i. e., if $\underline{\delta}_o' = (\delta_1 + \delta_2 + \ldots + \delta_p)$ then $||\delta_o||^2 = \delta_1^2 + \delta_2^2 + \ldots + \delta_p^2$.

Theorem 2

Let $\underline{\delta}_o(\lambda)$ denote the solution of equation (20) for a given value of λ. Then $|| \underline{\delta}_o(\lambda) ||^2$ is a continuous, decreasing function of λ such that, as $\lambda \to \infty$, $|| \underline{\delta}_o(\lambda) ||^2 \to 0$.

Theorem 3

Let γ denote the angle between $\underline{\delta}_o$ and δ_g. Then γ is a continuous monotone decreasing function of λ such that as $\lambda \to \infty$, $\gamma \to 0$. Since $\underline{\delta}_g$ is independent of λ, it follows that $\underline{\delta}_o(\lambda)$ rotates toward $\underline{\delta}_g$ as $\lambda \to \infty$.

If one observes that at $\lambda = 0$, equation (21) reduces to

$$\underline{P}_j' \underline{P}_j \underline{\delta}_o = \underline{P}_j'(Y - f_j) \qquad (24)$$

so that $\underline{\delta}_o(0)$ is equal to $\underline{\delta}_\ell$. Thus $\underline{\delta}_o$ is, as mentioned before, an interpolation between $\underline{\delta}_\ell$ (when $\lambda = 0$) and $\underline{\delta}_g$ (when $\lambda \to \infty$).

Marquart defines a scale of measurement for the coordinates of the parameter space. Since $\underline{\delta}_g$ is changed if the scales of the coordinates are changed unequally, Marquart uses scales in units of the quantities

$$c_{ii}^{\frac{1}{2}} = \left(\sum_{u=1}^{n} (\delta f_u(\theta)/\delta\theta_i)^2 \right)^{\frac{1}{2}} \tag{25}$$

This conversion causes the $\underline{P}_j' \underline{P}_j$ matrix to be rescaled into a matrix which has ones on the diagonal and numbers whose modulus is less than one as the off-diagonal elements. By defining

$$\underline{P}_j' \underline{P}_j = (C_{pq}^j) = \left(\sum_{u=1}^{n} \left(\frac{\delta f_u(\underline{\theta})}{\delta\theta_p} \right) \left(\frac{\delta f_u(\underline{\theta})}{\delta\theta_q} \right) \right) \tag{26}$$

One can evaluate a scaled matrix $\underline{A}_j$ and a scaled vector $\underline{G}_j$:

$$\underline{A}_j = (a_{pq}^j) = \frac{c_{pq}^j}{(c_{pp}^j \, c_{qq}^j)^{\frac{1}{2}}} , \tag{27}$$

$$\underline{g}_j = (g_{1j}, g_{2j}, \cdots, g_{pj}) = (g_{ij}) = \frac{\sum_{u=1}^{n} P_{iu}^j \, (Y_u - f_u(\underline{\theta}_j))}{(c_{ii}^j)^{\frac{1}{2}}} \tag{28}$$

After rescaling the units the new correction vector $\delta^*_{l_j}$ is obtained by evaluating

$$\underline{\delta}^*_{l} = \underline{A}_j^{-1} \, \underline{g}_j \tag{29}$$

If $\underline{A}j$ is singular we must solve

$$\underline{A}_j \, \underline{\delta}^*_{\ell j} = \underline{g}_j \tag{30}$$

Now, to obtain $\underline{\delta}_{\ell j}$, the original correction vector one must evaluate

$$(\underline{\delta}_{\ell j})_i = (\underline{\delta}^*_{\ell j})i/(c_{ii})^{\frac{1}{2}}, \; i = 1, 2, \ldots, p, \tag{31}$$

where $(\underline{\delta}_{\ell j})_i$ denotes the ith element of $\underline{\delta}_{\ell j}$. The major elements for implementation of the algorithm have now been assembled. At the jth iteration when $\underline{\theta}$ j becomes available (or is designated as on the first iteration), it is necessary to solve the equation

$$(\underline{A}_j + \lambda_j \, \underline{I}) \, \underline{\delta}^*_j = \underline{g}_j \tag{32}$$

for

$$\underline{\delta}^*_j = (\delta^*_{1j}, \delta^*_{2j}, \ldots, \delta^*_{pj})'$$

and evaluate

$$\delta_{ij} = \delta^*_{ij}/(c_{ii})^{\frac{1}{2}} \tag{33}$$

for i = 1, 2, . . ., p to obtain $\underline{\delta}$ j. A new vector is now defined by

$$\underline{\theta}_{j+1} = \underline{\theta}_j + \underline{\delta}_j \tag{34}$$

This vector must give a sum of squares $F(\underline{\theta}_{j+1})$ lower than the previous sum of squares $F(\underline{\theta}$ j). A value of λ_j must be chosen before solving equation (32) that will insure that

$$F(\underline{\theta}_{j+1}) < F(\underline{\theta}_j) \tag{35}$$

If a value of λ_j large enough cannot be obtained, then $F(\underline{\theta}_j)$ must be a minimum and $\underline{\theta}_j$ is the least squares solution vector.

A value for λ_j is needed that will satisfy equation (35) and will produce rapid convergence of the algorithm to the least squares solution for $\underline{\theta}$. The linearized function will adequately represent the non-linear function only over a small neighborhood. To produce rapid convergence, whenever conditions are such that linearization would work well by itself, a small value of λ_j should be chosen. The most important case of this is when $\underline{\theta}_j$ is close to the minimum as in the later iterations of the algorithm.

Marquart suggest a strategy for choosing λ_j (Marquart, 1963) which seems highly appropriate. It is as follows: choose an arbitrary number $v > 1$. Let λ_{j-1} denote the value of from the previous iteration. Initially let $\lambda_o = 0.01$. Compute $F(\lambda_{j-1})$ and $F(\lambda_{j-1}/v)$

i. If $F(\lambda_{j-1}/v) \leq F(\lambda_{j-1})$, let $\lambda_j = \lambda_{j-1}/v$.

ii. If $F(\lambda_{j-1}/v) > F(\lambda_j)$, and $F(\lambda_{j-1}) \leq F(\lambda_j)$, let $\lambda_j = \lambda_{j-1}$.

iii. If $F(\lambda_{j-1}/v) > F(\lambda_j)$, and $F(\lambda_{j-1}) > F(\lambda_j)$, increase by successive multiplication by v until for some smallest value w, $F(\lambda_j)$. Let $\lambda_{j-1} = \lambda_{j-1} \cdot v^w$.

Modifications to Marquart's Compromise

The Marquart Compromise Algorithm does not lend itself well to constrained problems. To overcome this shortcoming, a paper published in 1971 (Smith and Shanno, 1971) presents a modification which uses a slightly different gradient form (called a Newton-Raphson vector) to allow handling of constrained problems. Smith and Shanno also

designed their algorithm to conduct an analytic search for a locally optimum value of the combining parameter λ_j which they indicate has given excellent results.

Another modification to the original algorithm was suggested by Marquart himself (Marquart, 1970). The method is new in that it uses the generalized inverse as a numerical analysis tool in obtaining the least squares estimates for the parameters. The modified algorithm has much the same properties as the one based on ordinary inverses of $(A + \lambda I)$. Both methods interpolate between the linearization method and the steepest descent method. The new algorithm uses the scaled matrix $\underline{A}_j$ and the scaled vector g_j as defined in (27) and (28). The scaled equation

$$\underline{A}_j \underline{\delta}_j = \underline{g}_j \tag{31}$$

is then constructed. Let the diagonalized matrix be denoted $\underline{D}$ with ordered elements $K_1 \geq K_2 \ldots > K_p$, and the eigenvector matrix that transforms $\underline{A}$ into $\underline{D}$ be denoted $\underline{S}$. Thus,

$$\underline{S}' \underline{A} \underline{S} = \underline{D} \tag{32}$$

where $\underline{S}' \underline{S} = \underline{I}$, and

$$\underline{A}^{-1} = \underline{S} \underline{D}^{-1} \underline{S}' \tag{33}$$

If $\underline{A}$ is of rank r, then the last (p-r) elements of $\underline{D}$ are zeros if $\underline{A}$ is singular. $\underline{S}$ can be partitioned as follows:

$$\underline{S} = (\underline{S}_r \vdots \underline{S}_{p-r}) \tag{34}$$

where $\underline{S}_r$ is $\lfloor p \times r \rfloor$ and $\underline{S}_{p-r}$ is $\lfloor p \times (p-r) \rfloor$. The matrix $\underline{D}$ can be partitioned also so that the generalized inverse becomes

$$\underline{A}_r^{-1} = \underline{S}_r \, \underline{D}_r^{-1} \, \underline{S}_r' \tag{36}$$

This inverse is then used to compute the gradient $\underline{\delta}_j$ and the iteration continues exactly as in the original algorithm.

The generalized inverse modification is probably superior to the original algorithm for ill-conditioned problems because of the reduction in variance achieved by the small amount of bias in the generalized inverse. Marquart presents several theorems to illustrate this in his article (Marquart, 1970) and he also adds that this procedure is equivalent to the original algorithm when $\underline{D}_r$ is equal to $\underline{D}$.

CHAPTER III

A NEW ALGORITHM FOR NON-LINEAR REGRESSION

Development

The development of the current version of the Sequential Simplex Algorithm is easily traced. In 1975, G.E.P. Box suggested a procedure for the optimization and control of full-scale chemical processes called "Evolutionary Operation." (Box, 1957). Since then several of the basic concepts of Box have been modified to form a powerful algorithm for solving non-linear regression problems, and indeed many other optimization problems, without calculating derivatives. Since it was first introduced in 1957, the term "Evolutionary Operation" has been more widely interpreted than Box described. In this discussion the term will be held in its original restricted sense--mathematical optimization. Much of the development of the present method stems originally from a consideration of how Evolutionary Operation might be made automatic--a possibility suggested by Box in his article in 1957.

The next major step in the development of the algorithm was taken by W. Spendley, G. R. Hext, and F. R. Himsworth (Spendley et al., 1962) in their presentation of an article describing a basic sequential simplex optimization procedure. Their article was concerned to a large extent with the development of a sequential simplex method for automatic processing by a digital computer--

specifically the problems of when to move and where to move after the simplex was formed. The decision rule proposed was to move at every opportunity, that is, at every iteration. Since Spendley et al. were dealing with full-scale processes actually in operation, each iteration introduced a more recent observation. This reduced the errors caused by measurement drift and changes in the process and thus was a logical decision rule.

The question of "where to move" was also influenced by the idea of frequent changes in the simplex. A major attraction of the simplex design is that by adding one additional point and deleting an old point, an entirely new simplex is obtained in the hyperplane containing any face of the original simplex. Also, the direction of steepest descent estimated from the observations at the vertices of a regular simplex will be along a line drawn through the center of the simplex out through that face which is "opposite to" (does not contain) the point of highest value. Because of this, Spendley, Hext, and Himsworth decided that the answer to the "where to move" question was into the adjacent simplex which is obtained by discarding the point of highest value and adding its mirror image in the hyperplane of the remaining points.

The article states four major advantages of such a method. They are:

1. The computation is extremely simple. At each iteration, only one additional point must be found and its value calculated.
2. The direction of advance is dependent solely on the

ranking of responses and not on their values. Thus, the system can be used in cases where a numerical value cannot be absolutely determined, but can be determined in relation to the other responses.

3. No assumptions of the curvature of the response surface outside the immediate simplex is made.
4. To add additional factors, one point added for each factor is all that is required. Thus, if a factor previously held constant was now made variable, only one point must be added to a K dimensional simplex to make it one in (K+1) dimensions.

The idea of "regularity" of the simplex in the geometrical sense was deemed important by Spendley, Hext, and Himsworth. They proposed that regularity be preserved by choosing the scales of the separate factors so that a unit change in one factor is of equal interest with a unit change in another factor. This regularity was further obtained by using the procedure described above for all new points which were introduced into the simplex.

This rigid requirement served to slow the method considerably and frequently caused poor results for complicated surfaces. The regularity requirement and many other details of the method were changed in a subsequent modification by J. A. Nelder and R. Mead (Nelder and Mead, 1965). The procedure described by Nelder and Mead allows for changes in the shape and size of the simplex so that it adapts itself to the local landscape, elongating down long inclined planes and contracting in complex areas, especially in

the neighborhood of the minimum.

An algorithm based on a modification of the Nelder and Mead method is presented in the following section. A Fortran II computer program for this algorithm is given in Appendix I.

The Method

General Description

The basic experimental design employed is a simplex in G dimensions, where G is the number of parameters or factors to be determined. A simplex may be defined by a $(G+1)$ X G matrix $\underline{M}$, say

$$\underline{M} = \begin{bmatrix} 0 & 0 & 0 \ldots 0 \\ p & q & q \ldots q \\ q & p & q \ldots q \\ \cdot & \cdot & \cdot \quad \cdot \\ \cdot & \cdot & \cdot \quad \cdot \\ q & q & q \ldots p \end{bmatrix}$$

where

$$p = L\ (1/G\sqrt{2}\,)(\ (G-1)\ +\sqrt{G+1}\ ) \qquad (1)$$

$$Q = L(1/G\sqrt{2})(\sqrt{G+1} - 1) \qquad (2)$$

and L is the length desired for the sides of the original simplex. Once the scales have been determined and the origin defined, this matrix can be used as a convenient starting simplex. Its rows give the G coordinates of each of the $(G+1)$ vertices of the simplex.

A simplex is an orthogonal first-order experimental design

which requires the minimum number of experimental points in any number of dimensions; that is, it requires only one more experimental point than the number of variables under consideration. Therefore, for the G variables, there are

$$N = G+1 \tag{3}$$

design points (rows in the $\underline{M}$ matrix). The regular simplex has the N vertices so situated in the design space that the cosine of the angle formed by any two vertices and the center of the simplex is equal to -1/G. In two dimensions the regular simplex is an equilateral triangle, in three, a tetrahedron, and so on.

The responses are determined at each of the vertices. That is, for each set of coordinates in the design matrix, the value of the objective function is calculated. These values are then compared and the highest determined. The procedure then eliminates that set of coordinates (that vertex) from the simplex and calculates a new set of coordinates which mirror the one just eliminated in the hyperplane containing the other vertices of the simplex. The coordinates of this new vertex are calculated by the following method:

Call the current simplex matrix $\underline{M}_j$. Let the rows of $\underline{M}_j$ be denoted by $\underline{R}_i$ for $i = 1, 2, \ldots, G+1$. Let the maximum error sum of squares response occur at $\underline{R}_h$. Call the centroid of the remaining points $\underline{C}$ calculated by

$$\underline{C} = 1/G \; (\underline{R}_1 + \underline{R}_2 + \ldots \underline{R}_{h-1} + \underline{R}_{h+1} + \ldots + \underline{R}_{G+1}) \tag{4}$$

Then the coordinates of the new vertex $\underline{R}^*$ are given by

$$\underline{R}^* = \underline{C} + \lambda(\underline{C} - \underline{R}_h) \tag{5}$$

Where λ is the step length computed for this reflection. If $\lambda = 1$ then the new vertex is the mirror image of the one just removed from the simplex and is the vertex which would be chosen by the algorithm developed by Nelder and Mead (Nelder and Mead, 1965). However, the modified sequential simplex attempts to compute an optimal step length for each reflection. Define the error or differences $g_u(\underline{\theta})$ by

$$g_u(\underline{\theta}) = (Y_1 - f(\underline{X}_u, \underline{\theta}), \quad u = 1, 2, \ldots, m \tag{6}$$

where $\underline{\theta}$ is the vector of parameter estimates given by an vertex of the simplex. Each $g_u(\underline{\theta})$ is a non-linear function of the parameter $\underline{\theta}$. However, by assuming that the $g_u(\underline{\theta})$ are linear in the unknown parameter, one can estimate the optimal step length in the direction $\underline{d} = \underline{C} - \underline{R}_h$, which is the direction indicated by the simplex. Therefore, equation (5) can be rewritten as

$$\underline{R}^* = \underline{c} + \lambda\ \underline{d} \tag{7}$$

The error sum of squares at this new vertex is

$$F(\underline{R}^*) = F(\underline{c} + \lambda\underline{d}) \tag{8}$$

and since $\underline{C}$ and $\underline{d}$ are both known, it can be written as

$$F(R^*) = F(\lambda) \tag{9}$$

At the minimum of $F(\lambda)$ it is necessary that

$$\frac{\delta F(\lambda)}{\delta \lambda} = 0 \tag{10}$$

It follows from the above that

$$F(\lambda) = F(\underline{c} + \lambda \underline{d}) = \sum_{u=1}^{m} g_u(\underline{c} + \lambda \underline{d})^2 \tag{11}$$

and

$$\frac{\delta F(\lambda)}{\delta \lambda} = 2 \sum_{u=1}^{m} g_u(\underline{c} + \lambda \underline{d}) \frac{\delta g_u(\underline{c} + \lambda \underline{d})}{\delta \lambda} \tag{12}$$

From the assumption that each $g_u(\underline{c} + \lambda \underline{d})$ is linear, it is seen that

$$g_u(\underline{c} + \lambda \underline{d}) = g_u(\underline{c}) + g_u(\lambda \underline{d}) = g_u(\underline{c}) + \lambda (g_u(\underline{c}) - g_u(\underline{R}_h)) \tag{13}$$

Therefore

$$g_u(\underline{c} + \lambda \underline{d}) = g_u(\underline{c}) + \lambda\, h_u(\underline{c}, \underline{R}_h) \tag{14}$$

where

$$h_u(\underline{c}, \underline{R}_h) = g_u(c) - g_u(R_h) \tag{15}$$

and

$$\frac{\delta g_u(\underline{c} + \lambda \underline{d})}{\delta \lambda} = g_u(\underline{c}) - g_u(\underline{R}_h) = h_u(\underline{c}, \underline{R}_h) \tag{16}$$

Substituting (15) and (16) and (12) yields

$$2 \sum_{u=1}^{m} \left(g_u(\underline{c}) + \lambda\, h_u(\underline{c}, \underline{R}_h) \right) h_u(\underline{c}, \underline{R}_h) \tag{17}$$

Therefore

$$2 \sum_{u=1}^{m} g_u(\underline{c})\, h_u(\underline{c}, \underline{R}_h) = -\, 2\, \lambda \sum_{u=1}^{m} h_u(\underline{c}, \underline{R}_h)^2 \tag{18}$$

and

$$\tilde{\lambda} = - \frac{\sum_{u=1}^{m} g_u(\underline{c})\, h_u(\underline{c}, \underline{R}_h)}{\sum_{u=1}^{m} h_u(\underline{c}, \underline{R}_h)^2} \tag{19}$$

would be the optimal step length. Since it is known that the assumption of linearity in the unknown parameters is false, it is necessary to set a lower and upper bound for the step length. It is convenient to choose δ, the expansion coefficient, as the upper bound and α, the reflection coefficient as the lower bound. Therefore the step length used would be

$$\lambda = \min \left(\delta, \max \left(\tilde{\lambda}, \alpha \right) \right) \tag{20}$$

The algorithm must then insure that this simplex is in fact "better" than the old simplex in terms of getting closer to the minimum sum of squares. Since there are only two points not common to both, this check is merely a matter of comparing the responses at the two points.

If the new response is indeed lower than the old one and is also lower than the response at any other vertex, the algorithm then seeks to speed up the process by taking a larger step in that same

direction. This action is called an expansion, and if the simplex was regular before the move, the expansion destroys that regularity. The design is still a simplex, but it no longer has all sides equal.

If the response at the expanded point is lower than the mirrored (reflected) point then the expanded point is used as the new vertex of the simplex, if not the reflected point is used. Instances when the expanded point will not be lower occur when the response surface is complex and the reflected point is in a valley but the expanded point has been placed on the side where the response is higher.

If the reflected response is not lower than the highest sum of squares response of the old simplex then the algorithm takes a point half way between the centroid of the simplex and the reflected point. This step is called a contraction and is used to combat very complex surfaces and to insure proper termination of the algorithm. The termination criterion and procedure will be discussed later.

If the response at the contracted point is lower than the highest response of the old simplex, then the contracted point is used as the new point in the simplex replacing the vertex of the old simplex with the highest response. If the contracted point is not lower, then the algorithm forms a new simplex with only one point in common with the old one--that vertex with the lowest sum of squares response of all the vertices. This new simplex has sides one half the length of the old simplex. It is formed in this way to insure that progress is always made toward the lowest response. Also, a smaller simplex will have a better chance of progression

in situations which indicate a complex response surface or that the minimum has been reached. In either case, the smaller simplex will improve the chances of convergence.

Notation

The procedure may be better explained by the following notation:

R_i, $i = 1, 2, \ldots, G + 1$, are the $G + 1$ vertices of the original simplex.

F_i is the sum of squares function value at R_i.

h is the suffix such that $F_h = \max(F_i)$ $i = 1, 2, \ldots G + 1$

S is the suffix such that $F_S = \min(F_i)$ $i = 1, 2, \ldots, G + 1$

$\bar{R}$ is the centroid of the vertices where $i \neq h$.

$[R_i \; R_j]$ is the distance from R_i to R_j

λ is the estimated step length.

α is the reflection coefficient.

β is the contraction coefficient.

δ is the expansion coefficient.

At each iteration there are three possible steps--reflection, expansion and contraction. The reflection of R_h is denoted by R* and its coordinates are defined by

$$R^* = (1 + \lambda)\,\bar{R} - \lambda\, R_h \tag{21}$$

where λ is positive and $\delta \geq \lambda \geq \alpha$. Usually α is given a value of one. Thus, $[R^* \; \bar{R}] = \lambda\,[R_h \; \bar{R}]$. If F* is between F_h and F_S, then the algorithm replaces R_h with R* the new simplex is formed, and a new iteration started.

If $F^* < F_S$, i. e., if the reflection has produced a new minimum, we seek to improve this by expanding $\underline{R}^*$ to $\underline{R}^{**}$ by

$$\underline{R}^{**} = \delta \underline{R}^* + (1 - \delta)\overline{R} \tag{22}$$

By this, δ is the ratio of the expansion distance to the reflection distance, i. e., the ratio of $\lfloor \underline{R}^{**}\overline{R} \rfloor$ to $\lfloor \underline{R}^*\overline{R} \rfloor$. If $F^{**} < F^*$ then $\underline{R}_h$ is replaced by $\underline{R}^{**}$ and a new iteration is begun. If $F^{**} > F^*$, then $\underline{R}^*$ is the new vertex and the algorithm continues with a new iteration. This is called a failed expansion and will be discussed later.

If F^* is larger than all the F_i, $i \neq h$, then we wish to contract. We define F_h as the smallest of F^* and the old F_h and change R_h accordingly. We form a new point R^{**} such that

$$\underline{R}^{**} = \beta \underline{R}_h + (1 - \beta) \overline{R} \tag{23}$$

β is the ratio of $\lfloor \underline{R}^{**} \overline{R} \rfloor$ to $\lfloor \underline{R}_h \overline{R} \rfloor$ and lies between 0 and 1. This step contracts the simplex. $\underline{R}_h$ is now replaced by $\underline{R}^{**}$ unless $F^{**} > F_h$. In this case all points are replaced except $\underline{R}_S$. The points are replaced by points defined by the relation

$$\underline{R}_i = \tfrac{1}{2} (\underline{R}_i + \underline{R}_S) \text{ for } i \neq S \tag{24}$$

This procedure shrinks the entire simplex in the direction of the lowest point.

Maximum Age of Vertices

Another advantage of this size adjustment procedure occurs when some complexity of the response surface causes the simplex to oscillate

about some vertex or vertices. This may be easily detected because the computer program keeps a record of the number of iteration each vertex has been included in the simplex. This record is called the age of the vertex. When the age of any vertex exceeds a set limit, the algorithm uses the previously described shrinking procedure to eliminate the problem. The ages of the vertices are checked at each iteration.

The determination of the maximum age a vertex can achieve in normal progression was the subject of a study by Spendley, Hext and Himsworth (Spendley, et al., 1965). Their calculations led them to present a fitted equation which approximates the maximum age a vertex can attain for any number of parameters. That equation is:

$$\text{max age} = 1.65\ G + 0.05\ G^2 \tag{25}$$

where G is the number of parameters and $G + 1$ the number of vertices. This equation is only applicable where the simplex size is small when compared to any complex features of the response surface because it approximates the maximum number of iterations a vertex can remain unchanged before the simplex moves in the same direction it moved from before, This is not to imply it has oscillated back to a previous simplex; it means it has taken a step in the opposite direction of a recent step. This estimated maximum proved entirely sufficient in all problems encountered. For the problems presented here it was simplified to

$$\text{max age} = 2\ G - 1 \tag{26}$$

which fits perfectly for $G = 2, 3, \ldots, 7$.

Termination Criterion

The termination criterion used by the Modified Sequential Simplex Algorithm is one advanced by Nelder and Mead in their simplex development (Nelder and Mead, 1965). At each iteration, the algorithm compares the standard error of the values of the sum of squares formula at the vertices given by

$$\text{Error} = \sqrt{\sum_{i=1}^{G+1} (F_i - \overline{F})/(G + 1)} \tag{27}$$

where

$$\overline{F} = \sum_{i=1}^{G+1} F_i/(G + 1)$$

with some given value. If this error is below that value, the method stops. This criterion depends on two concepts--that the simplex will not become too small in relation to the curvature of the response surface before it reaches the minimum and that the simplex will shrink around the minimum. The first concept is insured by choosing an initial size of the simplex that is appropriate for the curvature of the response surface in the current problem and the second by the nature of the algorithm.

CHAPTER IV

PROBLEMS AND RESULTS

Sample Problems

Problem A

Source: (Hartley, 1961)

The six responses in this problem represent the yields of wheat corresponding to six rates of application of fertilizer, which, on a coded scale, are given the values $X_h = -5, -3, -1, 1, 3, 5$. The model here is the exponential law of diminishing returns:

$$Y = f(X; L, B, K) = L + B_e^{KX} \tag{1}$$

In this model there is only one input variable, X, but three parameters; namely, $\theta_1 = L$, the asmpto t ic yield for large rates of fertilizer application, $\theta_3 = K$, the exponential rate of response decrease and $\theta_2 = \beta$, defining the midpoint response (at $X = 0$) by $L + \beta$.

Problem B

Source: Lettau, H. H. and Davidson, B., Exploring the Atmosphere's First Mile, vol. 1, Pergamon Press, 1957, p. 332-336.

Under adiabatic conditions, the wind speed Y is given by the non-linear model

$$Y = \theta_1 \log(\theta_2 X + \theta_3) + \epsilon \tag{2}$$

where X = nominal height of anemometer (cm). Estimate θ_1, θ_2, and

Table 1. Data for Problem A

X	Y
-5	127
-3	151
-1	379
1	421
3	460
5	426

Table 2. Data for Problem B

X	Y
40	490.2
80	484.2
160	672.7
320	759.2
640	837.2

θ_3 from the following data.

Problem C

Source: Hunter, W. G. and Atkinson, A. C., "Planning Experiments for Fundamental Process Characterization," University of Wisconsin Technical Report No. 59, December, 1965.

A certain chemical reaction can be described by the non-linear model

$$Y = \exp\left(-\theta_1 X_1 \exp\left[-\theta_2(1/X_2 - 1/620)\right]\right) + \epsilon \qquad (3)$$

where θ_1 and θ_2 are the parameters to be estimated, Y is the fraction of original material remaining; X_1 is the reaction time in minutes, and X_2 is the temperature in degrees Kelvin. The data for this problem follows in Table 3.

Problem D

Source: Box, G. E. P., and Hunter, W. G. "Sequential Design of Experiments for Non-linear Models," I.B.M. Scientific Computing Symposium in Statistics, 1965, pp. 113-137.

A certain chemical reaction can be described by the non-linear model

$$Y = \theta_1 \theta_3 X_1/(1 + \theta_1 X_1 + \theta_2 X_2) + \epsilon \qquad (4)$$

where Y is the reaction rate, X_1 and X_2 are partial pressures of reactants and product respectively, θ_1 and θ_2 are absorbtion equilibrium constants for reactant and product, and θ_3 is the effective reaction rate constant. The data for problem D is shown in Table 4.

Problem E

Source: Marquart, D. W., I. B. M. Share Program No. 1428, 1963.

This is called the Overlapping Gaussian Problem. The model is

Table 3. Data for Problem C

X_1	X_2	Y
120	600	.904
60	600	.951
60	612	.883
120	612	.780
30	631	.786
45	631	.697
15	639	.887
90	639	.288
25	639	.708
60	639	.436
30	639	.660

Table 4. Data for Problem D

X_1	X_2	Y
1.0	1.0	.124
2.0	1.0	.205
1.0	2.0	.071
2.0	2.0	.127
0.1	0.0	.158
3.0	0.0	.549
0.2	0.0	.251
0.3	0.0	.311
3.0	0.8	.294

$$Y = \theta_1 \exp\left[-\tfrac{1}{2}\left(\frac{X - \theta_2}{\theta_3} \right)^2 \right] + \theta_4 \exp\left[-\tfrac{1}{2}\left(\frac{X - \theta_5}{\theta_6} \right)^2 \right] + \epsilon$$

The six parameters are estimated from the data in Table 5.

Comparison Criterion

The basis of comparison between Marquart's Compromise Algorithm and the Modified Sequential Simplex Algorithm is the amount of time a UNIVAC 1108 computer used to process the problem in the FORTRAN IV computing language. Marquart's Compromise Algorithm is available as an IBM SHARE Library Program No. 309401 NLIN 2, and the Modified Sequential Simplex Algorithm was written for this thesis.

A widely used criterion is the number of function evaluations necessary for solution, but this was not applicable in this case due to basic differences in the methods. Marquart's Compromise procedure uses relatively few function evaluations; yet, it must evaluate derivatives at each iteration. The Modified Sequential Simplex, on the other hand, evaluates no derivatives but depends solely on the determinations of the function for its method of progress.

Results

The performances of the two methods have been studied using five problems with 2, 3, and 6 parameters to estimate. As the results will depend on the initial starting point selected, each problem was solved using at least three different starting points and the results averaged. The starting values used for these problems are presented in Table 9 in Appendix II. The final values for the parameters are

Table 5. Data for Problem E

X	Y
0	-0.0145
0.1	-0.0104
0.2	0.0037
0.3	-0.0045
0.4	-0.0126
0.5	0.0400
0.6	0.1804
0.7	0.4023
0.8	0.6933
0.9	0.9920
1.0	1.2082
1.1	1.1856
1.2	1.0589
1.3	0.7545
1.4	0.6376
1.5	0.4891
1.6	0.4162
1.7	0.3038
1.8	0.1623
1.9	0.1430
2.0	0.0304
2.1	0.0705
2.2	-0.0271
2.3	-0.0153
2.4	0.0084
2.5	-0.0198

also listed in Appendix II in Table 10.

The sequential Simplex Method is also dependent on the size of the original simplex and upon the choice of the reflection, contraction and expansion coefficients. For the results given here, the Modified Sequential Simplex Algorithm was utilized with three combinations of α, β, and δ and with two different sizes of the original simplex. The data presented in Table 6 is the average of this detailed data. The detailed data is presented in a table in Appendix II.

From the data in Table 6, the algorithm by Marquart solved on the average all problems except C faster than the Modified Sequential Simplex Algorithm. Although no statistical tests were run to determine if the differences in solution times were significant, it appears that they are. No explanation has been found for the apparent success of the sequential simplex method in problem C; however, it is often the case that a particular method works well on certain isolated problems or classes of problems.

Care must be exercised to insure that for comparison purposes the advantage of one method over the other is not guaranteed by the choice of starting points. For example, if one is comparing the steepest descent and the linearization methods and all points are chosen near the true minimum, then the linearization method clearly has an advantage. The same care must be exercised in choosing the problems used for comparison and for the same reason--to insure equal advantage for each method.

Table 6. Solution Time (Milliseconds)

Problem	Data is in Milliseconds	
	Marquart's Method	Sequential Simplex
A	1018	1365
B	1457	1521
C	2661	2349
D	2103	2395
E	3120	3833

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

A new method for finding the minimum of a sum of squares function used in non-linear regression problems has been described and compared to the method devised by D. W. Marquart. The new method is called the Modified Sequential Simplex Algorithm. It has the advantages of not requiring the calculation of derivatives and also of being extremely easy to program due to its simplicity.

The Modified Sequential Simplex Algorithm has been shown to be only slightly slower than Marquart's Compromise Method, in general cases, and was even faster in some special problems. It is possible to provide for faster convergence than that of Marquart's method by the correct choice of simplex orientation, starting point, and step size, but these results are only special cases and can not be expected in general.

This method can be modified to include constrained optimization problems or any minimization problem, even if it is not a sum of squares. This is also true of most of the methods discussed in this thesis. The FORTRAN IV program, presented as an appendix to this thesis, may also be easily modified for these problems.

The parameter estimates obtained by the Modified Sequential Simplex method compare favorably with Marquart's procedure in terms of accuracy. They can be computed to any desired degree of numerical

accuracy. This is achieved by forcing the simplex to shrink tighter around the point on the response surface with the lowest sum of squares.

The Modified Sequential Simplex Algorithm also allows simple conversion from problem to problem, and all that must be changed is the representation of the function. To convert Marquart's algorithm, however, three FORTRAN subroutines must be rewritten each time the problem changes.

From the results of the problems investigated for this thesis, it is concluded that, if the first derivatives and second derivatives exist and are calculable, Marquart's Compromise Method is slightly superior to the Modified Sequential Simplex Method for most problems. However, it is felt that some improvement has been made over the algorithm presented by Nelder and Mead. There exist many other modifications which potentially could increase the speed of the algorithm.

Recommendations

This investigation only illustrates the feasibility of the Modified Sequential Simplex Method. It is recommended that the algorithm be further tested and possible modifications made and evaluated to obtain more efficient computation times. The loss of simplicity in the method can be offset by the increased speed with which it might operate, and therefore, even very complex modifications should be evaluated.

The assumption that the differences $Y_u - f(\underline{X}_u, \underline{\theta})$ are linear

in the parameters should be reconsidered for its possible modification to an assumption that they are more complex functions of $\underline{\theta}$, e. g. quadratic. This modification could possibly increase the speed of the algorithm significantly and should be evaluated. The choices of α, β and δ should also be studied with a larger group of problems to obtain a rational method for their selection.

APPENDIX I

```
****   THESIS    ***   MAIN PROGRAM    ***
DATE:  05 18 71  TIME:  174218


C       SEQUENTIAL SIMPLEX NONLINEAR REGRESSION ANALYSIS
C       M = NO. OF DATA POINTS
C       LSIZE = NO. OF VARIABLES
C       INDV = NO. OF INDEPENDENT VARIABLES IN MODEL
C       X(I,J) = DESIGN MATRIX
C       B(I) = COEFFICIENTS IN THE OBJECTIVE FUNCTION
C       MAX = MAXIMUM NUMBER OF ITERATIONS ALLOWED
C       SIDELN = SIDELENGTH OF THE INITIAL SIMPLEX
C       IAGE(I) = AGE OF SIMPLEX VERTEX
C       VALUE(I) = OBJECTIVE FUNCTION VALUE
C       OBJFUN = SUBROUTINE WHICH CALCULATES SUM OF SQUARES
C       FUN = FUNCTION WHICH SUPPLIES OBJECTIVE FUNCTION
C       ALPHA = REFLECTION
C       BETA = EXPANTION
C       GAMMA = CONTRACTION
        DIMENSION R(20)
        DIMENSION X(20,20), XNEW(20,20), IAGE(20), VALUE(20),SUM(20)
        DIMENSION B(20), DATA(30,30)
        COMMON/BLOK1/ALPHA,BETA,GAMMA
        COMMON/BLOK2/VALUE,N,M
        COMMON/BLOK3/X,SUM,INDEXL
        COMMON/BLOK4/DATA,INDV
C       READ IN FIRST COORDINATES AND FORM INITIAL SIMPLEX
        READ95,5000)M,INDV,LSIZE,MAX,SIDELN,ALPHA,BETA,GAMMA,EPSLON
 5000   FORMAT(4I4,5F10.0)
        READ(5,5001)(B(I),I=1,LSIZE)
 5001   FORMAT(10F8.0)
        N=LSIZE + 1
        ITER=0
        KNTR=1
        G=LSIZE
        LL=INDV + 1
        DO 30 I=1,M
   30   READ(5,5002)(DATA(I,J),J=1,LL)
 5002   FORMAT(5F8.0)
 1001   DO 1 I=1,N
    1   IAGE(I)=0
        DO 2 I=1,LSIZE
    2   X(1,I)=B(I)
        P=SIDELN*(SQRT(G+1.0)-1.0+G)/(G*SQRT(2.0))
        Q=SIDELN*(SQRT(G+1.0)-1.0)/(G*SQRT(2.0))
        DO 4 I=2,N
        DO 3 J=1,LSIZE
```

```
    3  X(I,J)=Q+B(J)
       X(I,KNTR)=P+B(KNTR)
    4  KNTR=KNTR+1
       DO 7 I=1,N
       DO 6 J=1,LSIZE
    6  B(J)=X(I,J)
       CALL OBJFUN(SS,B)
    7  VALUE(I)=SS
  707  DO 8 I=1,N
       DO 8 J=1,LSIZE
    8  XNEW(I,J)=X(I,J)
       ITER=ITER+1
       DO 9 I=1,N
    9  WRITE(6,6000)ITER,I,VALUE(I),(X(I,J),J=1,LSIZE)
 6000  FORMAT(1H,2I4,2X,E15.7,2X,10F10.4)
       DO 90 I=1,N
   90  IAGE(I)=IAGE(I)+1
       CALL CHECK(IAGE,LK)
       IF(LK.EQ.0)GO TO 919
       DO 918 I=1,LSIZE
  918  B(I)=X(1,I)
       WRITE(6,819)
  819  FORMAT(  '  LAGE EXCEEDED ')
       GO TO 1001
  919  CONTINUE
       IF(ITER.GT.MAX)GO TO 1000
       L=0
       CALL TERM(L,EPSLON)
       IF(L.EQ.1)GO TO 2000
C
C      FIND LARGEST VERTEX
C
       LARGE=VALUE(1)
       INDEXL=1
       DO 10 I=2,N
       IF(VALUE(I).LE.LARGE)GO TO 10
       LARGE=VALUE(I)
       INDEXL=I
   10  CONTINUE
C
C      FIND SMALLEST VERTEX
C
       SMALL=VALUE(1)
       INDEXS=1
       DO 11 I=2,N
       IF(VALUE(I).GE.SMALL)GO TO 11
       SMALL=VALUE(I)
       INDEXS=I
   11  CONTINUE
C
C      FIND CENTROID-DROP LARGEST VERTEX
```

```
C
      DO 13 I=1,LSIZE
      SUM(I)=0.0
      DO 12 J=1,N
      IF(J.EQ.INDEXL)GO TO 12
      SUM(I)=SUM(I)+X(J,I)
   12 CONTINUE
   13 SUM(I)=SUM(I)/G
C
C     DETERMINE STEP LENGTH
C
      CALL STEP(XLAMDA)
      DO 14 I=1,LSIZE
      B(I)=SUM(I)+XLAMDA*(SUM(I)-XNEW(INDEXL,I))
   14 CONTINUE
      CALL OBJFUN(SS,B)
C
C     IS REFLECTED POINT A NEW MIN OR MAX 4
C
      IF(SS.LT.SMALL)GO TO 17
      INDC=0
      DO 15 I=1,N
      IF(I.EQ.INDEXL)GO TO 15
      IF(SS.GE.VALUE(I))GO TO 15
      INDC=1
   15 CONTINUE
      IF(INDC.EQ.0)GO TO 21
C
C     IF WE GET HERE THE NEW VERTEX IN NEITHER MIN OR MAX
C
   18 VALUE(INDEXL)=SS
      DO 16 I=1,LSIZE
   16 X(INDEXL,I)=B(I)
      IAGE(INDEXL)=0
      GO TO 707
C
C     IF REFLECTION PRODUCES NEW MIN
C
   17 CONTINUE
      DO 19 I=1,LSIZE
   19 R(I)=SUM(I)+BETA*(SUM(I)-XNEW(INDEXL,I))
      CALL OBJFUN(SS1,R)
      IF(SS1.GT.VALUE(INDEXS))GO TO 18
      VALUE(INDEXL)=SS1
      DO 20 I=1,LSIZE
   20 X(INDEXL,I)=R(I)
      IAGE(INDEXL)=0
      WRITE(6,6060
 6060 FORMAT( ' ********EXPANSION******  ')
      GO TO 707
C
```

```
C       REFLECTED POINT IS A NEW MAX
C
   21   IF(SS.GT.VALUE(INDEXL))GO TO 23
        DO 22 I=1,LSIZE
   22   XNEW(INDEXL,I)=B(I)
        IAGE(INDEXL)=0
   23   DO 24 I=1,LSIZE
   24   B(I)=SUM(I)+GAMMA*(SUM(I)-XNEW(INDEXL,I))
        CALL OBJFUN(SS,B)
        WRITE(6,6061)
 6061   FORMAT( ' *****  CONTRACTION**  **')
        IF(SS.LE.LARGE)GO TO 18
C
C       IF WE GET HERE WE HAVE A FAILED CONTRACTION
C
        WRITE(6,9508)
 9508   FORMAT( ' ****      FAILED CONTRACTION  '  )
        DO 26 I=1,N
        IAGE(I)=0
        IF(I.EQ.INDEXS)GO TO 26
        DO 25 J=1,LSIZE
   25   X(I,J)=(X(I,J)+X(INDEX,J))/2.0
   26   CONTINUE
        DO 28 I=1,N
        DO 27 J=1,LSIZE
   27   B(J)=X(I,J)
        CALL OBJFUN(SS,B)
        IAGE(I)=0
   28   VALUE(I)=SS
        GO TO 707
 1000   BEST=VALUE(1)
        INDEXB=1
        DO 908 I=2,N
        IF(VALUE(I).GT.BEST)GO TO 908
        BEST=VALUE(I)
        INDEXB=I
  908   CONTINUE
        WRITE(6,6001)(X(INDEXB,I),I=1,LSIZE)
 6001   FORMAT(1H, 'MAX ITERATIONS EXCEEDED',/,10F10.4)
 2000   STOP
        END
```

- - - T H E E N D - - -

```
THESIS  ** OBJFUN SUBROUTINE
DATE:  051271  TIME:  115203


       SUBROUTINE OBJFUN(SS,B)
       COMMON/BLOK2/VALUE,N,M.
       COMMON/BLOK3/X,SUM,INDEXL
       COMMON/BLOK4/DATA,INDV
       DIMENSION B(20),VALUE(20),X(20,20),SUM(20),DATA(30,30)
       SS=0.0
       DO 10 I=1,M
       YHAT=FUN(B,I)
   10  SS=SS+(YHAT-DATA(I,INDV+1))**2
C          **********************
 9507  FORMAT( ' SS =  ',F10.4)
       RETURN
       END
```

- - - T H E E N D - - -

```
THESIS ** FUNCTION  FUNSUB (FUN)
DATE:  051271  TIME:  115548


       FUNCTION FUN(B,I)
       COMMON/BLOK4/DATA,INDV
       DIMENSION B(20),DATA(30,30)
C
C      PLACE MODLE HERE IN TERMS OF B(I) AND DATA(I,J)
       FUN=B(1)+B(2)+EXP(B(3)*DATA(I,1))
C
       RETURN
       END
```

- - - T H E E N D - - -

```
THESIS ** CHECK SUBROUTINE
DATE:  051271   TIME:  115930


      SUBROUTINE CHECK(IAGE,LK)
      COMMON/BLOK2/VALUE,N,M.
      DIMENSION VALUE(20)
      DIMENSION IAGE(20)
      LAGE=N*3
      LK=0
      DO 10 I=1,N
      IF(IAGE(I).LT.LAGE)GO TO 10
      LK=1
   10 CONTINUE
 9500 FORMAT( ' CHECK CALLED    LK = ',I4)
      RETURN
      END
```

- - - T H E E N D - - -

APPENDIX II

Table 7. Average Problem Solution Times for Simplex.

Problem	Starting Simplex #	(α, β, δ)	Milliseconds
A	1	$(1, \frac{1}{2}, 2)$	1373
		(1, 1/3, 3)	1368
		$(1, \frac{1}{2}, 4)$	1302
	2	$(1, \frac{1}{2}, 2)$	1587
		(1, 1/3, 3)	1289
		$(1, \frac{1}{2}, 4)$	1359
B	1	$(1, \frac{1}{2}, 2)$	1773
		(1, 1/3, 3)	1691
		$(1, \frac{1}{2}, 4)$	1638
	2	$(1, \frac{1}{2}, 2)$	1410
		(1, 1/3, 3)	1607
		$(1, \frac{1}{2}, 4)$	1013
C	1	$(1, \frac{1}{2}, 2)$	2043
		(1, 1/3, 3)	2989
		$(1, \frac{1}{2}, 4)$	1918
	2	$(1, \frac{1}{2}, 2)$	2738
		(1, 1/3, 3)	2141
		$(1, \frac{1}{2}, 4)$	2265
D	1	$(1, \frac{1}{2}, 2)$	2841
		(1, 1/3, 3)	2046
		$(1, \frac{1}{2}, 4)$	2983
	2	$(1, \frac{1}{2}, 2)$	2012
		(1, 1/3, 3)	2289
		$(1, \frac{1}{2}, 4)$	2162
E	1	$(1, \frac{1}{2}, 2)$	3139
		(1, 1/3, 3)	2977
		$(1, \frac{1}{2}, 4)$	2652
	2	$(1, \frac{1}{2}, 2)$	4683
		(1, 1/3, 3)	4921
		$(1, \frac{1}{2}, 4)$	4745

Table 8. Solution Times for Marquart's Method in Milliseconds.

Problem	Time for Solution
A	1192
	1095
	777
B	1424
	1391
	1556
C	2598
	2414
	2970
D	1875
	2036
	2398
E	2951
	3283
	3125

Table 9. Starting Points.

Starting Points	A	B	C	D	E
1	K=-.16 B=-180 L= 580	θ_1=130.6 θ_2^1= 2.12 θ_3^2=-34.3	θ_1= .004 θ_2^1=29,000	θ_1= 2.9 θ_2^1=12.2 θ_3^2= .69	b_1=0.7 b_1^2=0.9 b_2^3=0.15 b_3^4=0.7 b_4^5=1.2 b_5^6=0.4
2	K=-.18 B=-156 L= 590	θ_1=141.0 θ_2^1= 2.00 θ_3^2=-35.6	θ_1= .001 θ_2^1=26,000	θ_1= 3.1 θ_2^1=11.8 θ_3^2= .76	b_1=1.0 b_1^2=0.7 b_2^3=0.09 b_3^4=0.5 b_4^5=1.0 b_5^6=0.2
3	K=-.17 B=-201 L=570	θ_1=126.3 θ_2^1= 2.35 θ_3^2=-33.7	θ_1=.003 θ_2^1=29,500	θ_1= 4.9 θ_2^1=13.1 θ_3^2= .58	b_1=0.8 b_1^2=1.2 b_2^3=0.2 b_3^4=1.0 b_4^5=1.5 b_5^6=0.6

Table 10. Final Parameter Values.

Problem	Marquart	Simplex
A	$K = -.1994$ $B = -157.061$ $L = 523.350$	$K = -.1993$ $B = -157.063$ $L = 523.351$
B	$\theta_1 = 115.224$ $\theta_2 = 2.310$ $\theta_3 = -22.022$	$\theta_1 = 115.222$ $\theta_2 = 2.309$ $\theta_3 = -22.024$
C	$\theta_1 = .00376$ $\theta_2 = 27,539$	$\theta_1 = .00375$ $\theta_2 = 27,537$
D	$\theta_1 = 3.576$ $\theta_2 = 12.773$ $\theta_3 = 0.637$	$\theta_1 = 3.571$ $\theta_2 = 12.776$ $\theta_3 = 0.639$
E	$b_1 = 1.14077$ $b_2 = 1.0199$ $b_3 = 0.214$ $b_4 = 0.4013$ $b_5 = 1.4909$ $b_6 = .2606$	$b_1 = 1.14078$ $b_2 = 1.0197$ $b_3 = 0.215$ $b_4 = 0.411$ $b_5 = 1.4907$ $b_6 = .2603$

BIBLIOGRAPHY

REFERENCES CITED

Box, G. E. P., 1957, "Evolutionary Operation: A Method for Increasing Industrial Productivity," Applied Statistics, VI, pp. 3-22.

Fletcher, R. and M. J. D. Powell, 1963, "A Rapidly Convergent Descent Method for Minimization," Computer Journal, Vol. 6, pp. 163-168.

Hartley, H. O., 1961, "The Modified Gauss-Newton Method for Fitting Non-linear Regression Functions by Least Squares," Technometrics, Vol. 3, pp. 269-281.

Marquart, D. W., 1963, "An Algorithm for Least Squares Estimation of Non-linear Parameters," SIAM, Vol. 2, pp. 431-441.

Nelder, J. A., and R. Mead, 1965, "A Simplex Method for Function Minimization," Computer Journal, Vol. 7, pp. 308-313.

Powell, M. J. D., 1962, "An Iterative Method for Finding Stationary Values of a Function of Several Variables," Computer Journal, Vol. 5, p. 147.

Powell, M. J. D., 1965, "A Method for Minimizing a Sum of Squares of Non-linear Functions Without Calculating Derivatives," Computer Journal, Vol. 7, p. 303.

Ramsay, J. O., 1970, "A Family of Gradient Methods for Optimization," Computer Journal, Vol. 13, No. 4, pp. 413-417.

Smith, F. B. and D. F. Shanno, 1971, "An Improved Marquart Procedure for Non-linear Regression," Technometrics, Vol. 13, No. 1, p. 63-74.

Spendley, W., G. R. Hext, and F. R. Himsworth, 1962, "Sequential Application of Simplex Designs in Optimization and Evolutionary Operation," Technometrics, Vol. 4, pp. 441-461.

OTHER REFERENCES

Barnes, J. G. P., 1965, "An Algorithm for Solving Non-linear Equations based on the Secant Method," Computer Journal, Vol. 8, p. 66.

Box, G. E. P., 1957, "On Experimental Attainment of Optimal Conditions," Journal of the Royal Statistical Society, Vol. 13, No. 6.

Box, M. J., 1965, "A New Method of Constrained Optimization and a Comparison with other Methods," Computer Journal, Vol. 8, 1965, pp. 42-52.

Box, M. J., 1966, "A Comparison of Several Current Optimization Methods," Computer Journal, Vol. 9, p. 67.

Carrol, C. W., 1961, "The Created Response Surface Technique for Optimizing Non-linear Restrained Systems," Operations Research, Vol. 9, p. 169.

Davidon, W. C., 1959, "Variable Metric Method for Minimization," AEC Research and Development Report ANL-5990, p. 409.

Fletcher, R., 1965, "Function Minimization Without Evaluating Derivatives- A Review," Computer Journal, Vol. 7, p. 149.

Fletcher, R., and C. M. Reeves, 1964, "Function Minimization by Conjugate Gradients," Computer Journal, Vol. 7, p. 149.

Marquart, D. W., R. G. Bennett, and E. J. Burrell, 1961, "Least Squares Analysis of Electron Paramagnetic Resonance Spectra," Journal of Molecular Spectroscopy, Vol. 7, No. 4, pp. 269-279.

Mullen, C. J., 1970, "Simplex Evolutionary Operation for Improvement of Traffic Signal Settings," Master's Thesis, Georgia Institute of Technology, June, 1970.

Powell, M. J. D., 1965, "A Method for Minimizing a Sum of Squares of Non-linear Functions Without Calculating Derivatives," Computer Journal, Vol. 7, pp. 303-307.

Rosen, J. B., 1960, "The Gradient Projection Method for Non-linear Programming," SIAM, Vol. 8, p. 181.

Rosenbrock, H. H., 1960, "An Automatic Method for Finding the Greatest or Least Value of a Function," Computer Journal, Vol. 3, p. 175.

Spendley, W., 1967, "Non-linear Least Squares Fitting Using a Modified Simplex Minimization Method," Minimization, Ed. R. Fletcher, Academic Press, London, pp. 259-270.

Wilde, D. J., 1964, Optimum Seeking Methods, Prentice Hall, Englewood-Cliffs, N. J.

Wilde, D. J., 1967, Foundations of Optimization, Prentice Hall, Englewood Cliffs, New J.